Directory-Based Information Services: Users and Groups

Status of this Memo

Distribution of this memo is unlimited.

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on March 13, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.
Abstract

This document extends Directory-Based Information Services (DBIS) described in [draft-bannister-dbis-mapping-00] to support passwd and group databases.

The passwd and group database schemas SHALL be backwards compatible with the Network Information Service [NIS] but stored within [X.500] entries so that they may be resolved with the Lightweight Directory Access Protocol [RFC4510].

A passwd database represents user login accounts on UNIX and UNIX-like systems and a group database represents user groups.

This document describes configuration maps [draft-bannister-dbis-mapping-00] for passwd and group databases, and database entries referenced by those maps.

Overlays may optionally be used to help reduce the complexity of merging multiple DBIS domains in large environments by permitting groups of hosts to have variations in their UIDs, GIDs, home directories and login shells.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED" and "MAY" in this document are to be interpreted as described in [RFC2119].

Table of Contents

1. Configuration Maps ................................. 4
1.1. Scope ........................................ 4
1.2. Example Configuration Map Entries ................. 4
2. Database ......................................... 5
2.1. passwd ........................................ 5
2.1.1. Definition .................................... 5
2.1.2. Object Classes ............................... 5
2.1.2.1. Introduction ............................... 5
2.1.2.2. dbisPasswdConfig ......................... 6
2.1.2.3. posixUserAccount ......................... 6
2.1.3. Attributes .................................... 6
2.1.3.1. dbisMapGecos .............................. 6
2.1.3.2. dbisOverlayDN ............................. 7
2.1.3.3. en ....................................... 7
2.1.3.4. uidNumber ................................. 7
2.1.3.5. exactPrimary ............................... 7
2.1.3.6. homeDirectory ............................. 8
2.1.3.7. authPassword ............................... 8
2.1.3.8. userPassword ............................... 8
1. Configuration Maps

1.1. Scope

All databases described in this document use the standard configuration maps defined in [draft-bannister-dbis-mapping-00], section 3.

Additionally, dbisMapConfig entries for passwd and group databases SHALL have assigned the object classes dbisPasswdConfig and dbisGroupConfig respectively.

It is RECOMMENDED that the dbisMapConfig entry for a passwd or group database have the dbisMapFilter attribute set according to the following table:

<table>
<thead>
<tr>
<th>Database</th>
<th>dbisMapFilter</th>
</tr>
</thead>
<tbody>
<tr>
<td>passwd</td>
<td>objectClass=posixUserAccount</td>
</tr>
<tr>
<td>group</td>
<td>objectClass=posixGroupAccount</td>
</tr>
</tbody>
</table>

1.2. Example Configuration Map Entries

The following gives an example of a configuration map entry for a passwd database:

dn: cn=passwd, en=sales.corp, ou=domain-mappings, o=infra
objectClass: top
objectClass: dbisMapConfig
objectClass: dbisPasswdConfig
cn: passwd
dbisMapDN: cn=passwd, ou=dbis, o=infra
dbisMapFilter: objectClass=posixUserAccount
dbisMapGecos: displayName
profileTTL: 900
description: Primary passwd database
The following gives an example of a configuration map entry for a group database:

```
  dn: cn=group,ou=domain-mappings,o=infra
  objectClass: top
  objectClass: dbisMapConfig
  objectClass: dbisGroupConfig
  cn: group
  dbisMapDN: cn=group,ou=dbis,o=infra
  dbisMapFilter: objectClass=posixGroupAccount
  profileTTL: 900
  description: Primary group database
```

2. Database

2.1. passwd

2.1.1. Definition

A passwd database contains the following fields:

- User name.
- User password.
- Numeric user identifier (UID).
- Numeric group identifier (GID) of the user’s primary group.
- Full descriptive name of user (GECOS).
- Path to user’s home directory.
- Path to user’s login shell.

DBIS also adds the following information:

- Optional list of secondary groups of which the user is a member.

The information that makes up a database entry is obtained from the attributes described in the following sections.

2.1.2. Object Classes

2.1.2.1. Introduction

A dbisMapConfig entry for a passwd database SHALL be assigned the
object class dbisPasswdConfig.

A passwd entry SHALL be defined by an LDAP entry with the object class posixUserAccount. As this is an auxiliary class, it MUST also have a structural class assigned that is not defined in this document, for example inetOrgPerson [RFC2798].

2.1.2.2. dbisPasswdConfig

The dbisPasswdConfig class is defined as follows:

objectclass ( 1.3.6.1.4.1.23780.219.1.8 NAME 'dbisPasswdConfig' 
DESC 'DBIS passwd configuration map' 
SUP dbisMapConfig STRUCTURAL 
MUST dbisMapGecos 
MAY dbisOverlayDN )

2.1.2.3. posixUserAccount

The posixUserAccount class is defined as follows:

objectclass ( 1.3.6.1.4.1.23780.219.1.9 NAME 'posixUserAccount' 
DESC 'User account with POSIX attributes' 
SUP top AUXILIARY 
MUST ( en $ uidNumber $ exactPrimary $ homeDirectory ) 
MAY ( authPassword $ userPassword $ loginShell $ exactGroup $ exactNetgroup $ disableObject ) )

2.1.3. Attributes

2.1.3.1. dbisMapGecos

The "gecos" field traditionally holds the user’s full name and sometimes other descriptive information about the account, information that is better stored in more specifically named attributes. As there are a variety of ways of storing this information already available this document does not define an additional field for the gecos information, but rather the dbisMapGecos attribute that MUST be assigned to a dbisPasswdConfig entry and which holds the name of the attribute to use to provide gecos information. It is defined as follows:

attributetype ( 1.3.6.1.4.1.23780.219.2.13 NAME 'dbisMapGecos' 
DESC 'Source attribute for gecos field' 
EQUALITY caseIgnoreIA5Match 
SYNTAX 1.3.6.1.4.1.115.121.1.26 SINGLE-VALUE )

The posixUserAccount object class is auxiliary and must always be
associated with another structural class. One such class is inetOrgPerson [RFC2798]. If user accounts were given the inetOrgPerson class, then displayName might be an appropriate value for the dbisMapGecos attribute.

2.1.3.2. dbisOverlayDN

One or more DNs identifying the search base for overlay entries are stored in the dbisOverlayDN attribute that MAY be assigned to a dbisPasswdConfig entry:

attributetype ( 1.3.6.1.4.1.23780.219.2.14 NAME 'dbisOverlayDN' 
DESC 'DN of search base for DBIS overlay entries' 
EQUALITY distinguishedNameMatch 
SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )

Overlays are described in section 3 of this document.

2.1.3.3. en

The name of the user account is stored in the LDAP attribute en which is defined in [draft-bannister-dbis-mapping-00]. The en attribute MUST be associated with a posixUserAccount entry and SHALL form the RDN.

2.1.3.4. uidNumber

The UID is stored in the uidNumber attribute that MUST be assigned to a posixUserAccount entry:

attributetype ( 1.3.6.1.1.1.1.0 NAME 'uidNumber' 
DESC 'An integer uniquely identifying a user in an administrative domain' 
EQUALITY integerMatch 
ORDERING integerOrderingMatch 
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

2.1.3.5. exactPrimary

The primary group name is stored in the exactPrimary attribute that MUST be assigned to a posixGroupAccount entry:

attributetype ( 1.3.6.1.4.1.23780.219.2.15 NAME 'exactPrimary' 
DESC 'Name of primary group' 
EQUALITY caseExactMatch SINGLE-VALUE 
SUBSTR caseExactSubstringsMatch 
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768} )
When generating a NIS-compatible passwd database entry, a DUA must convert this attribute to the GID number in order to produce the corresponding primary GID field.

For compatibility, this attribute may alternatively contain a GID rather than a group name. This is intended to support existing configurations only and SHOULD NOT be used for new entries. A DUA MUST support both formats, and will treat the attribute as a GID if it contains digits only.

2.1.3.6. homeDirectory

The path to the user’s home directory is stored in the homeDirectory attribute that MUST be assigned to a posixUserAccount entry:

\begin{verbatim}
attributetype ( 1.3.6.1.1.1.1.3 NAME 'homeDirectory'
  DESC 'The absolute path to the home directory'
  EQUALITY caseExactIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
\end{verbatim}

2.1.3.7. authPassword

The user’s encrypted password is stored in the authPassword attribute which is defined in section 2.5 of [RFC3112] and that MAY be assigned to a posixUserAccount entry.

While a DUA MAY implement any authentication password scheme supported by the DSA, it MUST support the CRYPT scheme for backwards compatibility, which is an implementation of the traditional UNIX crypt algorithm. However, it is RECOMMENDED that a more secure scheme is used.

If the authPassword attribute has more than a single value, the DUA SHOULD select a password based on the strongest authentication scheme that it supports and use that for authentication. If the authentication fails, the DUA SHALL NOT attempt to use any other values. If the attribute does not use a scheme supported by the DUA, then the DUA SHALL NOT successfully authenticate.

If a posixUserAccount entry does not have an authPassword or userPassword attribute, then the account is locked. A DUA SHALL NOT successfully authenticate locked accounts.

Transfer of authPassword values is strongly discouraged where the underlying transport service cannot guarantee confidentiality and may result in disclosure of the values to unauthorised parties.

2.1.3.8. userPassword
For compatibility, the user’s encrypted password may alternatively be stored in the userPassword attribute which is defined in section 2.41 of [RFC4519] and that MAY be assigned to a posixUserAccount entry.

This is intended to support existing configurations only and SHOULD NOT be used for new entries, which should use authPassword instead. A DUA MUST support both formats, but SHALL ignore the userPassword attribute entirely if authPassword is provided for an account.

The string representation of the userPassword attribute SHALL match the following grammar, which is described in ABNF notation [RFC5234]. The productions used that are not defined below can be found in section 1.4 of [RFC4512]:

```plaintext
scheme       = "crypt" / "md5" / "sha" / "ssha" / altscheme
altscheme    = "x-" keystring
userPassword = LCURLY scheme RCURLY cryptpass
```

Where "cryptpass" referred to in the above grammar represents the password key hashed by the designated algorithm. If the scheme is "sha", then a SHA-1 digest of the password is computed, and the encrypted password shall be the base64 encoding of the result.

While a DUA MAY implement any authentication password scheme supported by the DSA, it MUST support the "crypt" scheme for backwards compatibility, which is an implementation of the traditional UNIX crypt algorithm. However, it is RECOMMENDED that a more secure scheme is used.

If the userPassword attribute has more than a single value, the DUA SHOULD select a password based on the strongest authentication scheme that it supports and use that for authentication. If the authentication fails, the DUA SHALL NOT attempt to use any other values. If the attribute does not use a scheme supported by the DUA, then the DUA SHALL NOT successfully authenticate.

See also the authPassword attribute in section 2.1.3.7.

2.1.3.9. loginShell

The path to the user’s login shell is stored in the loginShell attribute that MAY be assigned to a posixUserAccount entry:

```plaintext
attributetype ( 1.3.6.1.1.1.4 NAME 'loginShell'
  DESC 'The path to the login shell'
  EQUALITY caseExactIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
```
If the loginShell is missing, then this user will not be able to login to any host or service that requires a UNIX shell.

2.1.3.10. exactGroup

A list of one or more secondary group names are stored in exactGroup attributes that MAY be assigned to a posixGroupAccount entry:

attributetype ( 1.3.6.1.4.1.23780.219.2.16 NAME 'exactGroup'
DESC 'One or more secondary group names'
EQUALITY caseExactMatch
SUBSTR caseExactSubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768} )

This is an alternative to providing the exactUser attribute on a posixGroupAccount entry, as described in section 2.2.3.5 of this document.

2.1.3.11. exactNetgroup

The user can have netgroup membership expressed by providing netgroup names in one or more exactNetgroup attributes defined in [draft-bannister-dbis-netgroup-00] and that MAY be assigned to a posixUserAccount entry.

This attribute is provided as an alternative mechanism to using the netgroupUser attribute on the netgroupObject entry but with the limitation that the user will be considered a member of the netgroup regardless of which host or domain they are logged into. If the host or domain are important, the membership can only be expressed via the netgroupObject entry.

The DUA SHALL validate that a netgroup referenced by this attribute exists and is enabled. If the netgroup is not defined, or if it has been disabled with the disableObject attribute, then it SHALL NOT be included in the response to the client.

2.1.3.12. disableObject

A user account MAY be disabled by setting the disableObject attribute [draft-bannister-dbis-mapping-00] to TRUE. If an entry is disabled, then the DUA behave as if the user does not exist. The DUA MAY optionally provide a separate mechanism for listing disabled entries, but they MUST be clearly marked as disabled so that no confusion can arise.

2.1.4. Example Passwd Entry
The following is an example of a posixUserAccount entry in LDIF format [RFC2849]. As posixUserAccount is an auxiliary class, it has in this example been attached to an instance of inetOrgPerson [RFC2798]:

```
station: en=mark,ou=passwd,ou=sales,o=infra
objectClass: top
objectClass: inetOrgPerson
objectClass: posixUserAccount
cn: Mark
sn: Bannister
displayName: Bannister, Mark
en: mark
uidNumber: 101
exactPrimary: staff
homeDirectory: /home/mark
loginShell: /bin/bash
exactGroup: sales
exactGroup: dev
exactNetgroup: engineering
```

2.2. group

2.2.1. Definition

A group database contains the following fields:

- Group name.
- Group password.
- Numeric group identifier (GID).
- List of member accounts.

Additionally, DBIS adds support for nested groups.

2.2.2. Object Classes

2.2.2.1. Introduction

A dbisMapConfig entry for a group database SHALL be assigned the object class dbisGroupConfig.

A group entry SHALL be defined by an LDAP entry with the object class posixGroupAccount.

2.2.2.2. dbisGroupConfig
The dbisGroupConfig class is defined as follows:

```
objectclass ( 1.3.6.1.4.1.23780.219.1.11 NAME 'dbisGroupConfig'
  DESC 'DBIS group configuration map'
  SUP dbisMapConfig STRUCTURAL
  MAY dbisOverlayDN )
```

**2.2.2.3. posixGroupAccount**

The posixGroupAccount class is defined as follows:

```
objectclass ( 1.3.6.1.4.1.23780.219.1.12 NAME 'posixGroupAccount'
  DESC 'Group account with POSIX attributes'
  SUP top STRUCTURAL
  MUST ( en $ gidNumber )
  MAY ( authPassword $ userPassword $ exactUser $ exactGroup $ uniqueMember $ description $ manager $ disableObject )
```

**2.2.3. Attributes**

**2.2.3.1. en**

The name of the group account is stored in the LDAP attribute en which is defined in [draft-bannister-dbis-mapping-00]. The en attribute MUST be associated with a posixGroupAccount entry and SHALL form the RDN.

**2.2.3.2. gidNumber**

The GID is stored in the gidNumber attribute that MUST be assigned to a posixGroupAccount entry:

```
attributetype ( 1.3.6.1.1.1.1.1 NAME 'gidNumber'
  DESC 'An integer uniquely identifying a group in an administrative domain'
  EQUALITY integerMatch
  ORDERING integerOrderingMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )
```

**2.2.3.3. authPassword**

The group’s encrypted password is stored in the authPassword attribute which is defined in section 2.5 of [RFC3112] and that MAY be assigned to a posixGroupAccount entry.

All considerations relating to the authPassword attribute given in section 2.1.3.7 of this document apply equally to posixGroupAccount entries, with the exception that if a posixGroupAccount entry does
not have an authPassword attribute, then any user with the group listed in their exactGroup attribute may switch to the group without having to provide authentication. If an authPassword attribute is set, then the user must provide the correct password before the DUA will permit a group switch.

2.2.3.4. userPassword

For compatibility, the group’s encrypted password may alternatively be stored in the userPassword attribute which is defined in section 2.41 of [RFC4519] and that MAY be assigned to a posixGroupAccount entry.

This is intended to support existing configurations only and SHOULD NOT be used for new entries, which should use authPassword instead. A DUA MUST support both formats, but SHALL ignore the userPassword attribute entirely if authPassword is provided for an account.

All considerations relating to the userPassword attribute given in section 2.1.3.8 of this document apply equally to posixGroupAccount entries.

See also the authPassword attribute in section 2.2.3.3.

2.2.3.5. exactUser

A list of one or more user account names who are members of the group are stored in exactUser attributes that MAY be assigned to a posixGroupAccount entry:

attributetype ( 1.3.6.1.4.1.23780.219.2.26 NAME 'exactUser' DESC 'One or more user account names' EQUALITY caseExactMatch SUBSTR caseExactSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768} )

This is an alternative to providing the exactGroup attribute on a posixUserAccount entry, as described in section 2.1.3.10 of this document.

2.2.3.6. exactGroup

A list of one or more group account names who are members of this group are stored in exactGroup attributes, defined in section 2.1.3.10 of this document, that MAY be assigned to a posixGroupAccount entry. This allows groups to be nested. A DUA SHALL support nested groups.
If a user is not an explicit member of a posixGroupAccount, implicit membership needs to be determined by recursively examining each exactGroup attribute as the group may inherit members from other groups. To prevent infinite loops, a DUA SHALL NOT test any group object more than once during a single membership operation.

2.2.3.7. uniqueMember

For compatibility, group members may alternatively be stored in the uniqueMember attribute which is defined in section 2.40 of [RFC4519] and that MAY be assigned to a posixGroupAccount entry.

This is intended to support existing configurations only and SHOULD NOT be used for new entries, which should use exactUser and exactGroup instead. A DUA MUST support both formats.

Members referenced by the uniqueMember attribute SHALL be assumed to have been presented via the existing configuration map, even if they were located in a different base DN. The uniqueMember attribute is therefore not suitable for referencing users or groups that are defined with a different schema. The exactUser and exactGroup attributes do not suffer this problem.

2.2.3.8. description

The description attribute MAY be associated with a posixGroupAccount entry to provide an arbitrary description of the entry.

2.2.3.9. manager

The manager attribute MAY be associated with a posixGroupAccount entry to provide one or more DNs of the individuals, groups or systems that are responsible for maintaining the entry.

2.2.3.10. disableObject

A group account MAY be disabled by setting the disableObject attribute to TRUE. If an entry is disabled, then the DUA SHALL behave as if the group does not exist. The DUA MAY optionally provide a separate mechanism for listing disabled entries, but they MUST be clearly marked as disabled so that no confusion can arise.

2.2.4. Example Group Entry

The following is an example of a posixGroupAccount entry in LDIF format [RFC2849]:

```
dn: en=finance,ou=group,ou=sales,o=infra
```
3. Overlays

3.1. Definition

Overlays provide alternate passwd and group entries that may override UID, GID, home directory or login shells for groups of hosts that share a configuration map. This is helpful when merging two DBIS domains with overlapping IDs by allowing a period of transition when hosts and services from the origin domain may continue to use their original IDs and login shells. A DUA SHALL implement overlays.

Consider the example where UserA and UserB have UIDs 100 and 101 and login shell /bin/sh on HostA and HostB, but need UIDs 1000 and 1001 and login shell /bin/csh on HostC and HostD. All four hosts are a member of the same DBIS domain. Overlays permit this type of configuration.

3.2. Object Classes

3.2.1. Introduction

The top-level DN underneath which to search for overlay entries SHALL be defined by the dbisOverlayDN attribute which is associated with either a dbisPasswdConfig or dbisGroupConfig entry. Overlay entries MUST reside underneath this DN if they are to be used by a DUA.

Overlay entries for the passwd database are identified by the object class dbisPasswdOverlay. Overlay entries for the group database have the class dbisGroupOverlay.

3.2.2. dbisPasswdOverlay

The dbisPasswdOverlay class is defined as follows:

```
objectclass ( 1.3.6.1.4.1.23780.219.1.13 NAME 'dbisPasswdOverlay'
  DESC 'User account overlay entry'
  SUP top STRUCTURAL
  MUST en
```

3.3. Attributes

3.3.1. en

The en attribute MUST be assigned to a dbisPasswdOverlay or dbisGroupOverlay entry and will be used to identify the corresponding posixUserAccount or posixGroupAccount entry to overlay. If the en attributes match exactly, or if this is a dbisPasswdOverlay and there is no exact match but a default entry exists identified by an en attribute containing a single asterisk (*), then the attributes provided in the overlay SHALL replace those in the original database entry.

When a DUA looks up a posixUserAccount or posixGroupAccount entry that has an overlay configuration, it SHALL also search for a dbisPasswdOverlay or dbisGroupOverlay entry.

If a default entry, i.e. en=*, is found, the uidNumber attribute is ignored if assigned to the dbisPasswdOverlay object.

3.3.2. uidNumber

An alternative UID to use for a matching user account is stored in the uidNumber attribute (see section 2.1.3.4) which MAY be associated with a dbisPasswdOverlay entry.

3.3.3. gidNumber

An alternative GID to use for a matching group account is stored in the gidNumber attribute (see section 2.2.3.2) which MAY be associated with a dbisGroupOverlay entry.

3.3.4. homeDirectory

An alternative home directory to use for a matching user account is
stored in the homeDirectory attribute (see section 2.1.3.6) which MAY be associated with a dbisPasswdOverlay entry.

3.3.5. loginShell

An alternative login shell to use for a matching user account is stored in the loginShell attribute (see section 2.1.3.9) which MAY be associated with a dbisPasswdOverlay entry.

3.3.6. disableObject

An overlay entry MAY be disabled by setting the disableObject attribute to TRUE. If an entry is disabled, then the DUA SHALL behave as if the overlay does not exist. The DUA MAY optionally provide a separate mechanism for listing disabled entries, but they MUST be clearly marked as disabled so that no confusion can arise.

3.3.7. Example Overlay Entries

The following is an example of a dbisPasswdOverlay entry in LDIF format [RFC2849], and corresponding dbisMapConfig entries. In this example, the user "julie" who logs into hosts that are part of the "sales-merger" netgroup will get an alternative UID of 5001 and "/bin/sh" as the login shell. If "julie" logs into any other host, she will get her normal UID and login shell:

```
dn: cn=passwd,en=sales.corp,ou=domain-mappings,o=infra
objectClass: top
objectClass: dbisMapConfig
objectClass: dbisPasswdConfig
cn: passwd
dbisMapDN: cn=passwd,ou=dbis,o=infra
dbisMapFilter: objectClass=posixUserAccount
dbisMapGecos: displayName
notNetgroup: sales-merger
profileTTL: 900
description: Primary passwd database
```

```
dn: cn=passwd2,en=sales.corp,ou=domain-mappings,o=infra
objectClass: top
objectClass: dbisMapConfig
objectClass: dbisPasswdConfig
cn: passwd2
dbisMapDN: cn=passwd,ou=dbis,o=infra
dbisMapFilter: objectClass=posixUserAccount
dbisMapGecos: displayName
dbisOverlayDN: ou=passwd,ou=overlays,ou=sales-merger,o=infra
exactNetgroup: sales-merger
```
The following is an example of a dbisGroupOverlay entry which modifies the GID for the "finance" group when used in a configuration map entry:

```
dn: en=finance,ou=group,ou=overlays,ou=sales-merger,o=infra
objectClass: top
objectClass: dbisGroupOverlay
en: finance
gidNumber: 7308
```

4. Attribute Syntax

The following syntaxes are used by the attributes defined in this document:

```
<table>
<thead>
<tr>
<th>Syntax OID</th>
<th>Value</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.3.6.1.4.1.1466.115.121.1.12</td>
<td>DN</td>
<td>[RFC4517]</td>
</tr>
<tr>
<td>1.3.6.1.4.1.1466.115.121.1.15</td>
<td>Directory String</td>
<td>[RFC4517]</td>
</tr>
<tr>
<td>1.3.6.1.4.1.1466.115.121.1.26</td>
<td>IA5 String</td>
<td>[RFC4517]</td>
</tr>
<tr>
<td>1.3.6.1.4.1.1466.115.121.1.27</td>
<td>Integer</td>
<td>[RFC4517]</td>
</tr>
</tbody>
</table>
```

5. Implementation Notes

5.1. NIS Compatible Field Mapping

5.1.1. Introduction

All fields that are required to generate NIS-compatible colon-separated passwd or group database formats exist in this schema and can be mapped to attribute types using common ABNF productions described in [draft-bannister-dbis-netgroup-00], section 1.2.

These are described for each database in the following sections.

5.1.2. passwd
The NIS-compatible passwd database fields are mapped as follows:

```
user        = en
password    = %x78           ; lowercase "x", see below
uid         = uidNumber
gid         = gidNumber      ; derived, see below
gecos       = dbisMapGecos   ; derived, see below
homedir     = homeDirectory
loginshell  = loginShell
```

`passwd-entry = user COLON password COLON uid COLON gid COLON gecos COLON homedir COLON loginshell`

In the passwd mappings above:

- password is "x" which traditionally signifies that the password is actually stored in the shadow database. However, this was introduced historically as the shadow file could have stricter read permissions than the passwd file which would make the password more secure. As this is not relevant for an LDAP schema, the authPassword attribute is associated with the posixUserAccount object class. An implementer may therefore optionally report the encrypted password in NIS-compatible passwd entries, or not at all. For security it is RECOMMENDED that any individual user cannot display the encrypted password for any other user or group account, but only their own encrypted password. See section 6 for security considerations.

- gidNumber must be derived by a search for a posixGroupAccount entry that matches the name given in exactPrimary. An example search filter to achieve this can be found in section 5.2.4 of this document.

- gecos is determined by looking up the attribute identified by the dbisMapGecos attribute given on the configuration map entry. See section 2.1.3.1.

5.1.3. group

The NIS-compatible group database fields are mapped as follows:

```
  group    = en
password  = %x78           ; lowercase "x", see below
gid      = gidNumber
users    = exactUser      ; derived, see below
```

`group-entry = group COLON password COLON gid COLON users`
In the group mappings above:

- For security it is RECOMMENDED that the password be set to "x" as in section 5.1.2 and not to authPassword. See section 6 for security considerations.

- The list of users is a de-duplicated comma-separated list of exactUser attributes include those derived recursively through nested groups identified by the exactGroup attribute. See section 2.2.3.6.

5.2. Common Search Filters

5.2.1. Search Parameters

This section provides example LDAP search filters [RFC4515] for obtaining database entries with commonly used input criteria.

To simplify the examples, all databases are assumed to have been defined with only a single configuration map entry (dbisMapConfig). However, [draft-bannister-dbis-mapping-00] permits multiple such entries, so an implementation must support this, increasing the number of search operations as necessary to locate all of the database entries in scope.

The base DN used in the search operations described in this section comes from the dbisMapDN attribute assigned to the dbisMapConfig entry. Note that a dbisMapConfig entry may have more than one of these.

Where it appears in search filters below, the text "dbisMapFilter" refers to the value assigned to the attribute of the same name in the corresponding dbisMapConfig entry. Note that passwd and group databases have different dbisMapConfig entries. Attribute names used in these search filters may be modified by the dbisMapAttr attribute assigned to the dbisMapConfig entry.

5.2.2. Find Configuration Map for Domain

To locate the configuration map for a given DBIS domain, search for entries underneath the dbisDomainObject entry [draft-bannister-dbis-mapping-00].

Passwd maps can be found with the following search filter:

```ldap
(&(objectClass=dbisPasswdConfig)(!(disableObject=TRUE)))
```

Group maps can be found with:
5.2.3. List All Entries

Passwd and group entries are enumerated by applying the dbisMapFilter as follows:

\((\& (dbisMapFilter)(!(disableObject=TRUE)))\)

This filter returns all enabled entries.

5.2.4. Find Specific Entry

If a passwd or group entry is known by "name", its definition is located using the following search filter:

\((\& (dbisMapFilter)(!(disableObject=TRUE))(en=name))\)

5.2.5. Find Entry by ID

If a passwd entry has the UID "uid", its definition is located using the following search filter:

\((\& (dbisMapFilter)(!(disableObject=TRUE))(uidNumber=uid))\)

If a group entry has the GID "gid", it may be located using:

\((\& (dbisMapFilter)(!(disableObject=TRUE))(gidNumber=gid))\)

5.2.6. Find Netgroups By Membership

To obtain a list of all netgroups that a user with the login name "user" is a member of, the search filter from [draft-bannister-dbis-netgroup-00] section 6.4.5 is augmented to include all enabled netgroups listed in the exactNetgroup attribute on the user’s passwd entry, which is found using the search filter in section 5.2.4 above.

One additional search will be required to determine from a list of given netgroups which ones are enabled, as described in [draft-bannister-dbis-netgroup-00] section 6.4.7.

5.2.7. Member of a Specific Netgroup

To determine if a user with the login name "user" is a member of a specific netgroup called "name", the search filter from [draft-bannister-dbis-netgroup-00] section 6.4.6 is augmented to include a search for the exactNetgroup attribute on the user’s passwd entry:
6. Security Considerations

Passwd and group database entries contain encrypted passwords and SHOULD be transmitted securely when transferred between DSA and DUA to prevent eavesdropping. A DUA SHOULD NOT allow a user to see any encrypted passwords except they MAY see the password on their own posixUserAccount entry in encrypted form.

The security considerations discussed in [draft-bannister-dbis-mapping-00] and [RFC3112] apply equally to this document.

7. References

7.1. Normative References


7.2. Informative References


Author’s Address

Mark R. Bannister
Prose Consulting Ltd.
73 Claygate Lane
Esher, Surrey, KT10 0BQ
United Kingdom

Tel: +44 7764 604316
EMail: dbis@proseconsulting.co.uk